DOCUMENT RESUME

ED 161 421 IR 006 261

AUTHOR Gooch, Sherwin
TITLE PLATO Music Systems.
PUB DATE Mar 78
NOTE 17p.; Paper presented at the Annual Meeting of the Association for the Development of Computer Based Instructional Systems (Dallas, Texas, March 1-4, 1978); For related document, see IR 006 231

EDRS PRICE MF-$0.83 Plus Postage. HC Not Available from EDRS.
DESCRIPTORS *Computer Assisted Instruction; Computer Programs; Instructional Design; *Instructional Systems; *Musical Composition; *Music Education; On Line Systems
IDENTIFIERS *Instructional Development; Interactive Systems; Microprocessing; *PLATO

ABSTRACT

The original PLATO music concept was to replace the human performer in the feedback process, wherein the composer specifies an action and monitors the outcome, with a computer-controlled device. The first device of this type is known as the Gooch Synthetic Woodwind (GSW), which attempted to provide some of the features needed in an interactive, compositional facility, and many of those needed for computer-based music instruction. In the case of GSW, a PLATO-compatible music language was developed, as well as a compiler for this language, two music text editors, a filing system for music binaries, programs to play the music binaries in real time, and many debugging and compositional aids. A number of interactive compositional programs have also been written. With the advent of microprocessor technology, new PLATO terminals were developed to be less expensive and more flexible than the PLATO IV terminals. The 8080 microcomputer system in PLATO V terminals is capable of executing programs locally; it can also be connected directly to terminal peripheral devices. The goal of this system is to provide tools for music educators to use in the development of instructional materials, which might possibly include music dictation drills, automatically graded keyboard performances, envelope and timbre ear-training, interactive examples or labs in musical acoustics, and composition and theory exercises with immediate feedback. (VT)

****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

ED161421

U S DEPARTMENT OF HEALTH,
EDUCATION & WELFARE
NATIONAL INSTITUTE OF
EDUCATION

THIS DOCUMENT HAS BEEN REPRODUCED EXACTLY AS RECEIVED FROM THE PERSON OR ORGANIZATION ORIGINATING IT. POINTS OF VIEW OR OPINIONS STATED DO NOT NECESSARILY REPRESENT OFFICIAL NATIONAL INSTITUTE OF EDUCATION POSITION OR POLICY

PLATO MUSIC SYSTEMS

by

Sherwin Gooch

"PERMISSION TO REPRODUCE THIS MATERIAL IN MICROFICHE ONLY HAS BEEN GRANTED BY

John J. Hirschbuhl

TO THE EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC) AND USERS OF THE ERIC SYSTEM."

IR000261

PLATO MUSIC SYSTEMS

by Sherwin Gooch
Computer-based Education Research Laboratory
University of Illinois at Urbana-Champaign

The PLATO music project is rooted in a conjecture which is perhaps best stated in this manner: Johann Sebastian Bach may have been one of the greatest composers of all time due, in part, to his magnificent performing abilities. Performing ability is important in the original training of a composer's ear, and in the later maturation of this skill into the ability to "visualize" form in aural space. Performing ability is also of critical importance when the composer acts as a "passive observer", listening to his own performance of a composition, with the goal of later improving it. This is a feedback process wherein the composer specifies an action, and monitors the outcome. He then modifies the specifications, and repeats the feedback process until, if the labor is fruitful, the process converges to some preconceived form, and the piece is complete.

The original PLATO music concept was to replace the human performer in the feedback loop described above with a computer-controlled device. The first device of this type which was developed is known as the Gooch Synthetic Woodwind (GSW). The GSW represents a rudimentary attempt to provide, at minimum cost, some of the features needed in an interactive compositional facility, as well as many of those needed for computer-based music instruction.

The GSW provides nearly instantaneous control over the pitch and amplitude of four independent voices. (A voice is the electronic equivalent of a human voice in that it reproduces a single line of music.) Eight levels of volume are provided, which is sufficient to separate voices into solo and accompaniment, although it severely limits attempts to employ dynamics in a composition. The increments in pitch which the computer may specify are small enough to facilitate study of alternative tunings, including Just intonation in rapidly changing keys, and scales containing other than twelve tones.

Even within these limitations, this design philosophy seems to have met with some success. At the time of this writing, there are over fifty of these devices in use. They are being used for interactive ear-training drill, music dictation exercises, and in some compositions[1]. They also seem to have fostered a new artistic medium, which we refer to as "interactive art". In interactive art, the actions of the observer, sitting at a PLATO terminal, influence the performance of the composition. In the past, such experiences have been available only to conductors and solo performers.

Controlling even a device which is as conceptually simple as a GSW can be a horrendous programming task when the control is established through an interactive graphics computer system that is in a state of evolutionary flux. We estimate that only around ten percent of the human effort expended in the PLATO music project has gone into hardware-related activities; the other ninety percent of the effort has been in activities related directly to the generation and maintenance of software. The PLATO Music Group has concentrated its efforts on generating only the most basic programs needed to interface to music hardware, often providing only bare-bones examples of how courseware may be written, and leaving the monumental task of generating the actual PLATO courseware to the experts in the field.

At each evolutionary step, we try to provide what the PLATO community refers to as "author helps". This means that we provide easily-accessible, on-line, in-depth documentation of music hardware and software, as well as programs and program segments which will aid a PLATO author in writing courseware which utilizes PLATO music devices. In the case of the GSW, we developed a PLATO-compatible music language, a compiler for this language, two music text editors, a filing system for music binaries, and programs to play the music binaries in real time, as well as many debugging and compositional aids. The resulting subsystem is such that a PLATO author may type in a segment of music in a high level language, and have it played for his students by essentially "calling" a routine with the name that s/he gave the segment when the compiler stored the music binary, and the name of the "dataset" in which the binary is located. See figure one for an example of the source code for the language which has been developed.

--- *** --- INSERT FIGURE 1 HERE --- *** ---

The music compiler and related facilities are based on the assumptions most often encountered in "standard" western music. The form of the language and its default cases have been chosen so that typing in music from a score is as easy as possible. Some of the assumptions which were made are:

1) The fundamental element of the language is a "note".
2) In the default case, a note is determined by specifying a pitch and a duration.
3) In the default case, a pitch is specified by an octave number followed by the note's letter name. The default pitch is derived from the twelve-tone even-tempered scale.
4) Music flows at a tempo which is held constant until explicitly changed.
5) The relative location of the notes is topologically invariant with respect to standard music notation. This means that if you tilt your head ninety degrees to the right, the spatial location of a note with respect to any neighboring note is the same as it would be on a grand staff.

My philosophy has always been to make it as easy as possible for those unfamiliar with computers to accomplish the tasks which they most often desire to perform, but never at the expense of depriving the sophisticated user of the tools s/he may need. Following are a few examples of the tools which have been provided for the sophisticated user of the music compiler:

1) The user may specify an algorithm which will be used to compute the pitch operands to be stored in the note table, thereby allowing compositions to be written in non-standard tunings.
2) Arithmetic and algebraic expressions may be used to compute desired results in the source code.
3) The results of these expressions may be used to specify pitch in three different ways. The result of an expression may be interpreted as a note number (an index into the computable note table of example (1) above), it may be interpreted as the frequency in Hertz, or it may be used directly as an operand for the device. (The last has been found to be particularly useful in writing engineering test routines.)

4) Compile-time random number generation facilities are included.
5) The duration of a note may be specified in any one of four ways, demonstrated here by example:
   a) 3c4 quarter note
   b) 3c4(3) quarter note triplet
   c) 3c(.5) half note (may be an expression)
   d) 3c{6} six PLATO output frames (17.5 mS/frame)
6) Execution-time branching has been provided.
7) Complete macro facilities have been provided.

This last point deserves more comment. In many instances, a piece of music contains a recurring form. When this is the case, as in the example of Bach's Prélude in C from Das Wohltemperiertes Klavier (see fig. 2), one desires to specify only the information which changes from one repetition of the pattern to the next. Macros came into existance in computer languages for precisely this reason. Note that in the example, first the body of the macro is defined, setting the form for all following uses. When the macro is called repeatedly, only the changing arguments, which in this case are notes, need be specified. Therefore, each time the macro is called, the user is saved from the agony of having to type the static part of the form again. In addition to the labor-saving quality of macros shown here, we found that macros are extremely important from a compositional standpoint in that they provide a convenient shorthand for specifying a microform. Imagine how simply the pattern for a figured or alberti bass may be represented! (Only the pedal note need be given.) Since macros are expanded by character-string substitution, a minimum of restrictions are associated with their use.

--- *** --- INSERT FIGURE 2 HERE --- *** ---

In addition to the music compiler language which we developed, a number of interactive compositional programs have been written. One allows the user to set up a Schoenberg-style twelve-tone matrix. Compositions are generated by specifying each line as any transposition of the original tone row, its retrograde, inversion, or retrograde-inversion. Another is a simulation of what is known commercially as a "sequencer". Another generates an infinitely repeating loop of random note clusters of the length specified. Still other programs have been written which play pseudo-random notes while generating a display, a kind of multi-media artwork.

Two of the more important programs are called "gswfcn" and "gswrose". We feel that they are important because they portend what is to come in the future of real-time computer-based composition. They are also some of the earliest examples of interactive art. These programs generate sounds in which the pitches and volumes are determined by abstract mathematical equations which are functions of time. When one uses these "lessons", one specifies a set of equations which will determine the pitch and volume of each voice at each instant in time, and then listens to the sounds the equations generate. One may then modify the equations and listen again. If, at some point, there is a particular set of equations which one likes, one may name them as one does a composition, and store them to be played at a later time. After these equations are stored, they may also be used by any other interactive artist at any of the other one thousand PLATO terminals on the University of Illinois PLATO system as a starting point for her or his composition.

There are a number of remarkable things about this medium. One is simply that it is interactive and evolves with time. Another is that the human being serves more in the role of editor than of creator. The person who specifies the original equations may or may not understand precisely how the computer evaluation of the function will vary with time, but this is unimportant. What is important is that, in the case of all but the simplest (and most boring) functions, s/he has no idea of how the functions will sound. One just listens, and if one likes what one hears, the composition is saved. The last thing which is remarkable about this medium has to do with what some people in the artistic community have begun to refer to as the coming "American Renaissance" (which should, perhaps more properly, be referred to as the "American Naissance"), which is based on a coming-together of art and technology. One of the possibilities that advancing technology makes available to the artist of today is the ability to work in curvilinear (or continuously-variable) form. The coming age of curvilinear form is perhaps observed most easily in architecture. Architects are no longer restricted to designs containing lines, points, and [right] angles. Computational technology has given us the ability to build self-standing curvilinear structures; domes, arches, hanging structures, etc.

In music, one works in different spaces, those of pitch and time. The analog of a straight line or corner in these spaces is quantization. Historically, the number of pitches has been quantized by the notation and instruments available. Timing, once a pulse is established, has also been

quantized. The abstraction of mathematics, through the medium of computer-evaluated time-varying functions which describe pitch and amplitude, has given us a performable continuously-variable music notation.

To date we have used only PLATO-IV computer systems to support interactive real-time music synthesis. Conversion to any other computer system would be nearly impossible, due primarily to the vast arsenal of resources which are available on PLATO on which the music subsystem depends. The PLATO-IV system is a large interactive graphics computer system which is used primarily for computer-based education[2,3,4,5,6]. The PLATO system at the Computer-based Education Research Laboratory (CERL) is supported by two Control Data Corporation (CDC) Cyber 73 computers[7] with two million words of Extended Core Storage (ECS)[8], and supports over one thousand PLATO terminals which are located primarily in the United States. Each PLATO terminal[9] has the ability to deliver information to, or receive information from peripheral devices, which include audio disks[10], slide selectors[11], touch panels[12], joysticks, Morse code teaching devices, typewriters (which this paper was typed on), copiers (which the figures were made on), computer systems, floppy disk drives, speech synthesizers, music keyboards, music synthesizers, and many more.

The GSW, which was the first PLATO-compatible music synthesis device, was originally designed for use with the PLATO-IV terminal. With the advent of microprocessor technology, a new generation of PLATO terminals were developed. The new PLATO-V terminals, designed by Jack Stifle, embody all of the features of the PLATO-IV terminals at a lower cost[13], and also afford much greater flexibility. These terminals contain a complete microcomputer system which handles all of the logic and control functions of the terminal. For this reason, the PLATO-V terminal may be thought of as being a computer simulation of a PLATO-IV terminal. PLATO-V terminals are manufactured by Carroll Manufacturing of Champaign, Illinois.

The 8080 microcomputer system in the PLATO-V terminals is capable of executing programs locally. It can also be connected directly to terminal-peripheral devices. These two facts make it possible to accomplish a tremendous number of things related to interactive music synthesis which are not possible with the PLATO-IV terminal. The new functions which this terminal architecture makes possible can be divided into three categories, which are coding, buffering, and local processing.

Some of these enhancements are best clarified by example: Fifteen-bit pitch operands are delivered to the GSW over a data channel with an information bandwidth of twelve hundred bits per second at a rate sufficient to support four

voices. If we restrict ourselves to a universe of one hundred twenty eight notes, we can support eight voices with the same data channel, because we can fit two seven-bit note operands in the same space that is currently required for one fifteen-bit pitch operand. The problem then becomes one of coding; How would the note operands be converted to pitch operands? In this case, the answer is that a program may be loaded into the microcomputer of the PLATO-V terminal which performs the proper conversion. In actual practice, a more complex coding scheme may be adopted which will fit even more note operands into the same bandwidth, and also will preserve the ability to play music in alternate tunings.

Buffering techniques may be applied to both output from the central computer to the terminal (forward channel) and input to the central computer from the terminal (reverse channel). An example of reverse channel buffering occurs when using a polyphonic music keyboard connected to a PLATO terminal. Using the keyboard, one may easily play as many as ten notes per second. Assuming that timing information must be sent along with each note played on the music keyboard, as many as four words (sometimes called "keys") will have to be sent to the central PLATO computer for each note. This corresponds to an input rate of forty words per second. Unfortunately, the PLATO system was only designed to handle an average of two and one half words per second per terminal. How can we send forty words per second to a system which will only accept two and one half words per second? The answer is that we can save the information temporarily in the memory of the PLATO-V terminal as it is generated, and send it in to the central PLATO computer at a much lower rate, a technique known as buffering.

An example of local processing may be described in which the same music keyboard input device discussed above and a microcomputer-controlled music synthesizer are attached to the PLATO-V terminal. It may be desirable to have the synthesizer reproduce what is being played on the music keyboard while simultaneously displaying a standard score of the music as it is being performed. To accomplish this, a program can be written and loaded into the PLATO-V terminal which scans the music keyboard to see which keys are being depressed, sends the appropriate operands to the synthesizer, and also scores the music in real-time. All of this processing can be performed locally by the microcomputer contained in the PLATO-V terminal.

The Gooch Cybernetic Synthesizer (GCS) is a device which utilizes all three of these techniques.

The complete GCS hardware system is composed of two completely independent functional subunits which may be used individually if desired. One of the subunits is an intelligent music keyboard. This device contains a microprocessor which scans the music keyboard and alerts the microprocessor in the PLATO-V terminal to any changes in the state of the keyboard. The other subunit is the GCS itself. It is a computer-controlled polyphonic music synthesizer. There are plans to incorporate a microprocessor into the GCS in order to free more time in the PLATO-V terminal microprocessor for generating displays. This may also make it possible to use the synthesizer with other than PLATO-V terminals.

The envelope and waveform of each of the sixteen voices of the GCS are specified via program control. The envelope of a note is actually a function which specifies the instantaneous loudness of a note for the duration of time the note is played. The envelope consists of attack, decay, sustain, and release components (ADSR). During the attack (A) portion of the envelope, which is initiated when a key is depressed, the sound level is increasing. In the decay (D) portion, which immediately follows the attack, the sound level decreases to the sustain (S) level. The sustain portion of the envelope lasts until the key is released. During this time, the volume level of the note remains constant. The release (R) portion of the envelope is the time during which the sound level decreases to zero after the key has been released. In commercially available analog synthesizers, an envelope is designed by specifying four parameters, usually by setting knobs to the length of time that the A, D, and R portions will take, and setting a constant amplitude for the S portion.

In the most commonly used GCS envelope design program, one specifies attack, sustain, and release <u>functions</u>. The most popular way to describe these functions is by drawing graphs of the functions on a PLATO screen with a cursor editor. Since these are functions, and not single linear segments as in the case of analog synthesizers, one may design envelopes of a much greater complexity. Note that the decay portion of the envelope immediately follows the attack portion, so it may also be described by the attack function. The number of envelope generating functions is therefore reduced to three, attack, sustain, and release. In our system, the sustain function is repeated endlessly until the key is released. This makes it easy to generate an electronic tremolo effect. Oddities such as multiple attacks or releases are also easily accomplished.

The description of a key being depressed and released is employed in the above GCS description only as an explanatory paradigm. These functions may well be initiated via the GCS music keyboard, or they may be the result of computer control, or both. A program which exists at the time of this writing allows the user to play along with a computer rendition of a piece.

Ohm's Law of Acoustics states that, to a good approximation, the timbre of a sound is determined by its spectrum[14]. For this reason, the GCS waveform editor allows for drawing the frequency domain description of the waveform. The waveform editor then generates the Fourier transformation of this spectrum, yielding a time domain representation of the spectrum under consideration, which is the corresponding waveform. This waveform is then loaded into the GCS, where it is used to reproduce the desired audio spectrum. Upon hearing the result of this design, the composer is given the choice of storing the waveform in the computer system for later recall, or altering the spectral specifications and repeating the process until reaching the desired timbre.

--- *** --- INSERT FIGURE 3 HERE --- *** ---

At the time of this writing, software evolution is, as always, in progress. Our group has plans to provide facilities for the GCS which are, in each case, analogous to those which now exist for the GSW, and in addition:

1) A facility for generating music source from the music keyboard.
2) A graphical computer music language which is based not on character strings, but on standard music staves.
3) Interactive graphics editors for making edits to music written in the above language.
4) Envelope and waveform libraries. When the user desires that a certain synthetic instrument play a particular part, s/he need only call for it by name. In addition, PLATO has an analog-to-digital conversion input facility, and the parameters which best approximate real instruments will also be available from these libraries.
5) A keyboard practice program in which PLATO displays a score for the student to play. In this particular program, PLATO will not allow the student to progress until the student plays the correct notes.

6) A hard-copy music printing facility. Such a facility has been implemented which provides up to sixteen music staves per page. Sample output from this set of programs is available.

7) There are plans to provide a facility which will automatically assign multiple voices to an instrument in the foreground if fewer than the full sixteen voices are required for a given passage. This will not only improve the definition of a solo instrument, but will also provide a means for generating notes containing temporal evolution of the spectrum, a quality which is usually considered to be limited to synthesis by frequency modulation[15].

--- *** --- INSERT FIGURE 4 HERE --- *** ---

In summary, our goal is to provide tools for music educators to use in the development of instructional materials. Some of the possible instructional materials might include:

1) Music dictation drills involving up to sixteen voices. The variety of drill in which the student must find the difference between the score and that which is heard, as well as a new variety in which the student should actually be able to write on PLATO's music staff the music which has been played. PLATO could grade the dictation for accuracy, define the student's weaknesses, and generate the proper drill to correct the weaknesses.
2) Automatically graded keyboard performances.
3) Envelope and timbre ear-training.
4) Interactive examples or labs in musical acoustics.
5) Composition and theory exercises with immediate feedback.

In closing, I would like to leave you with this thought: Now, with the advent of digital storage techniques, for the first time in the history of art, it is possible to create a work of art which will survive without deterioration, exactly as it was created, for as long as there is rational life.

REFERENCES

1. Fred T. Hofstetter, "Music Dream Machines: New Realities for Computer-based Musical Instruction," Creative Computing, 3, No. 2 (1977), 50-54
2. Bruce Sherwood, The TUTOR Language, (Minneapolis, Minn.: Control Data Education Company, 1977)
3. D. L. Bitzer and D. Skaperdas, The Design of an Economically Viable Large Scaled Computer-based Education System, CERL Report X-5 (Urbana, Ill.: CERL, 1968)
4. J. E. Stifle, The PLATO IV Architecture, CERL Report X-20 (Urbana, Ill.: CERL,1971)
5. E. R. Lyman, A Summary of PLATO Curriculum and Research Materials, CERL Report X-23 (Urbana, Ill.: CERL, 1972)
6. P. T. Tucker, A Large Scale Terminal Controller, CERL Report X-27 (Urbana, Ill.: CERL, 1971)
7. Control Data Cyber 70 Model 73 Computer System (Minneapolis, Minn.: Control Data Corp., 1973)
8. Control Data Cyber 70 Computer System 7030 Extended Core Storage (Minneapolis, Minn.: Control Data Corp., 1971)
9. J. E. Stifle, The PLATO IV Student Terminal, CERL Report X-15 (Urbana, Ill.: CERL, 1977)
10. D. L. Bitzer, R. L. Johnson, and D. Skaperdas, A Digitally Addressable Random-Access Image Selector and Random-Access Audio System, CERL Report X-13 (Urbana, Ill.:CERL, 1970)
11. Ibid., p. 8.
12. F. Ebeling, R. S. Goldhor, and R. L. Johnson, "A Scanned Infrared Light Beam Touch Entry System," Digest of Papers - 1972 International Symposium, Society for Information Display (May 1971).
13. J. E. Stifle, The PLATO V Terminal, CERL Report X-50 (Urbana, Ill.: CERL, 1977)
14. M. R. Schroeder, "Models of Hearing," Proceedings of the Institute of Electrical and Electronics Engineers, 63, No. 9 (Sept. 1975), p. 1332
15. J. M. Chowning, "The Synthesis of Complex Audio Spectra by Means of Frequency Modulation," Journal of the Audio Engineering Society, 21, No. 7 (1973), 526-534

```
      BLOCK 1-d = inv 15                    SPACE =  26
 1  tempo    428.6,16   $$ 107 quarter notes per minute
 2  voice    2
 3  title       Inventio 15   by J. S. Bach
 4  title    Arr. for PLATO by Marshall Midden
 5  *
 6  key      d
 7  *
 8  measure 1
 9  1b8,r8
10  r8,3b16
11  ,3a+16
12  2d8,3b8
13  r8,3f8
14  2e8,t(3a,3g,3)8
15  r8,3f8
16  2d8,3b8
17  r8,3f8
18  *
19  measure 2
20  2c8,t(3g,3f,3)8
21  r8,3e8
22  1a+8,4c8
23  r8,3e8
24  1b8,t(3f,3e,3)8
25  r8,3d16
26  ,3c16
27  1g8,3d16
28  ,3e16
29  r8,3f16
30  ,3e16
31  *
```

Figure 1. This is an example of the music compiler language which was developed for use with the CSW. The "$t(x,y,n_1)n_2$" construction is an ornament. ("t" stands for "trill".)

```
      BLOCK 1-f = prelude 1                SPACE = 220
 1  title   Prelude in C,  by J. S. Bach
 2  tempo   285.7,16     $$ 286 sixteenth notes per minute
 3  voice   3
 4  *
 5  macro   form,A,B,C,D,E   $$ repeats twice per measure
 6  A2,r16,r8
 7    ,B8d,
 8    ,   ,C16
 9    ,   ,D16
10    ,B4t,E16
11    ,   ,C16
12    ,   ,D16
13    ,   ,E16
14  endm
15  macro   motif,A,B,C,D,E  $$ same for all but last 3 measures
16  form    A,B,C,D,E
17  form    A,B,C,D,E
18  endm
19  *
20  measure 1          $$ C
21  motif   3c,3e,3g,4c,4e
22  measure 2          $$ Dm7
23  motif   3c,3d,3a,4d,4f
24  measure 3          $$ G7
25  motif   2b,3d,3g,4d,4f
26  measure 4          $$ C
27  motif   3c,3e,3g,4c,4e
28  measure 5          $$ Am
29  motif   3c,3e,3a,4e,4a
30  measure 6          $$ D7
31  motif   3c,3d,3f+,3a,4d
```

Figure 2. This is an example of the use of macros in the GSW music compiler language. Macro "form" is nested inside macro "motif" so that each time "motif" is called, "form" is expanded twice. In this abstract form, each measure requires only one line of typing.

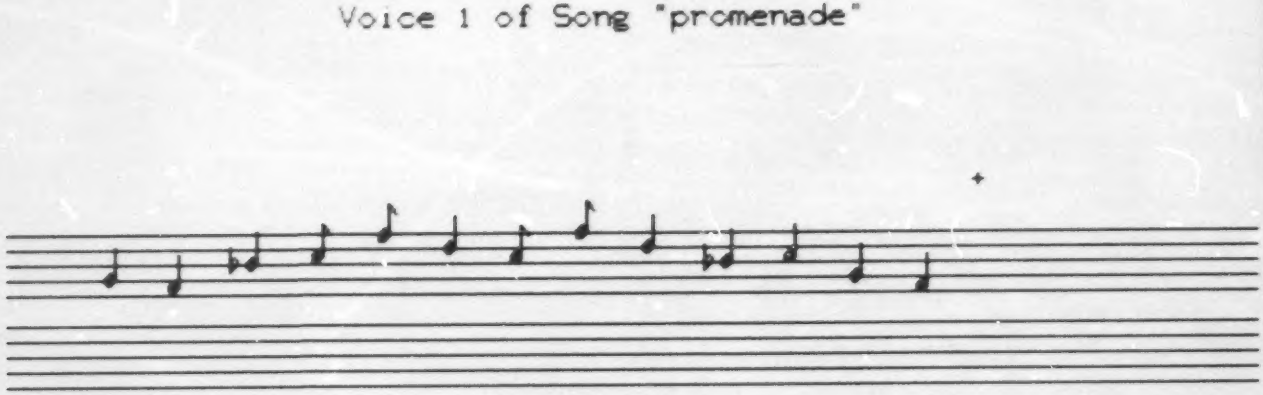

Figure 3. This is a display generated by a preliminary GCS "staff editor". To use the editor, one locates a position on the grand staff with the joystick-controlled cursor ("+" in the upper right), and then either adds or erases a note at that position.
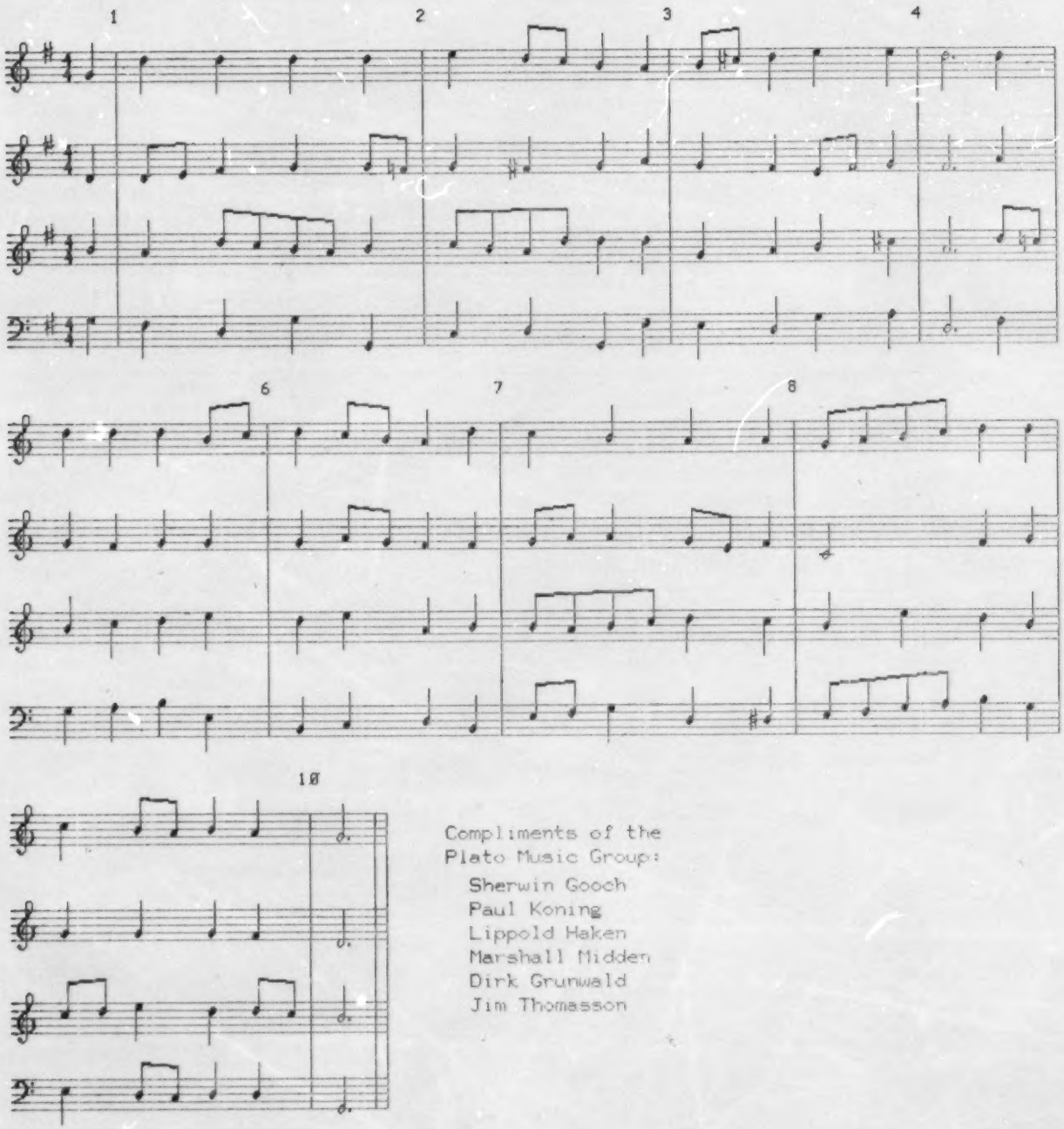
Figure 4. This is a sample of the output which can be generated by the hard-copy facilities. The musical notes used for this listing were played on the intelligent keyboard, then analyzed and sent to the PLATO computer by a preliminary "insert play" program.